



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/039,254	01/02/2002	Roni Rosner	042390.P12485	9368

8791 7590 11/27/2006

BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030

EXAMINER

KISS, ERIC B

ART UNIT PAPER NUMBER

2192

DATE MAILED: 11/27/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No. 10/039,254	Applicant(s) ROSNER ET AL.	
	Examiner Eric B. Kiss	Art Unit 2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 05 September 2006.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 30-43,45 and 47-66 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 30-43,45 and 47-66 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
 2. ☐ Certified copies of the priority documents have been received in Application No. _____.
 3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
- * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) <input type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | 5) <input type="checkbox"/> Notice of Informal Patent Application |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. The reply filed September 5, 2006, has been received and entered. Claims 30-43, 45, and 47-66 are pending.

Response to Amendment

2. Applicant's amendments to the claims appropriately address the objection to claims 35 and 51. Accordingly, this objection is withdrawn.
3. Applicant's amendments to claims 30 and 56 appropriately address the rejection of claims 30-32 and 56-61 under 35 U.S.C. § 101. Accordingly, this rejection is withdrawn.

Response to Arguments

4. Applicant's arguments filed September 5, 2006, have been fully considered but they are not persuasive.

Borrill teaches the use of instruction set architecture execution flags (an ISA tag) indicating the native ISA for "visiting" code (see, for example, col. 2, line 58, through col. 3, line 11; and col. 4, lines 30-58). *Borrill* teaches multiple ISA execution flags. A tag is read for each of a plurality of instructions (see *Borrill*, col. 4, lines 30-62). Further, the function of the Dynamic Decode Unit (DDU) of *Borrill* is to translate non-native instructions (see *Borrill*, col. 5, lines 19-30). *Borrill* further teaches the instruction set architecture execution flags being set by a programming environment of the binary (see, for example, col. 2, line 58, through col. 3, line 11; and col. 4, lines 30-58). The ISA flags of *Borrill* are used to control the translation of non-native code to a native format through an appropriate compatibility level, *i.e.*, an appropriate DDU or emulation (see, e.g., col. 5, lines 1-7).

Yates teaches that in the case of self-modifying code, these instructions are not translated because the execution is in general not predictable (Yates, col. 10, lines 44-53). However, the Borrill teachings further address the problems associated with self-modifying code, since all instructions are executed almost as native instructions are, by first being converted in real time by the DDU's, any code modifications are automatically accommodated, with no recompilation or other overhead encountered by binary translation or emulation (Borrill, col. 7, lines 25-32).

Claim Rejections - 35 USC § 103

5. The text of those sections of Title 35, U.S. Code not included in this action can be found in a prior Office action.

6. Claims 30-38, 40-43, 45, and 47-66 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No 5,802,373 to Yates et al. in view of U.S. Patent No. 6,496,922 to Borrill.

As per claims 30, 31, and 66, *Yates et al.* disclose receiving a binary of a program code, the binary based on a first instruction set architecture (see, for example, the Abstract);

translating the binary to a translated binary, wherein the translated binary is based on a combination of the first instruction set architecture and a second instruction set architecture (see, for example, the Abstract);

deviating from precise semantics of the binary during said translating in exchange for advantages offered by the second instruction set architecture (the original instructions are

translated into different instructions for execution on a different platform which has the advantages of being, among other things, newer and faster; see, for example, col. 1, lines 50-60); and executing the translated binary (see, for example, the Abstract).

The translated binary of *Yates et al.* is **based on** a combination of the first instruction set architecture and a second instruction set architecture.. As the pre-translation instructions were designed for the first instruction set architecture, the resulting post-translation instructions are **based on** (or derived from) those instructions and thus, **based on** (or derived from) the first instruction set architecture. Additionally, the translated code is **based on** the second (native) instruction set architecture, as this is a form that is directly executable (see *Yates et al.*, Abstract).

Yates et al. fail to disclose checking settable controls that have been set by a programming environment and that control said translating/deviating. However, *Borrill* teaches the use of instruction set architecture execution flags (an ISA tag) indicating the native ISA for “visiting” code (see, for example, col. 2, line 58, through col. 3, line 11; and col. 4, lines 30-58). *Borrill* teaches multiple ISA execution flags. A tag is read for each of a plurality of instructions (see *Borrill*, col. 4, lines 30-62). Further, the function of the Dynamic Decode Unit (DDU) of *Borrill* is to translate non-native instructions (see *Borrill*, col. 5, lines 19-30). *Borrill* further teaches the instruction set architecture execution flags being set by a programming environment of the binary (see, for example, col. 2, line 58, through col. 3, line 11; and col. 4, lines 30-58). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the method of *Yates et al.* to include such checking of settable controls (instruction set architecture execution flags) as per the teachings of *Borrill*. One would be motivated to do so to reduce overhead associated with translating and processing non-native

instructions and facilitate easier incorporation of non-native instructions into executable code by assigning meaningful instruction identifiers recognizable by the native system as suggested by *Borrill* (see, e.g., *Borrill*, col. 2, line 58, through col. 3, line 11).

As per claim 32, *Yates et al.* further disclose the translating of the binary comprising storing a portion of a hardware stack in a register of a processor translating the binary (see, for example, col. 46, lines 57-67).

As per claims 33 and 35, *Yates et al.* disclose receiving a binary of a program code, the binary based on a first instruction set architecture (see, for example, the Abstract); translating the binary, wherein the translated binary is based on a combination of the first instruction set architecture and a second instruction set architecture (see, for example, the Abstract); and executing the translated binary (see, for example, the Abstract).

The translated binary of *Yates et al.* is **based on** a combination of the first instruction set architecture and a second instruction set architecture. As the pre-translation instructions were designed for the first instruction set architecture, the resulting post-translation instructions are **based on** (or derived from) those instructions and thus, **based on** (or derived from) the first instruction set architecture. Additionally, the translated code is **based on** the second (native) instruction set architecture, as this is a form that is directly executable (see *Yates et al.*, Abstract).

Yates et al. fail to disclose checking settable controls that have been set by a programming environment and that indicate a compatibility level with which to perform the translation. However, *Borrill* teaches the use of instruction set architecture execution flags (an ISA tag) indicating the native ISA for “visiting” code (see, for example, col. 2, line 58, through col. 3, line 11; and col. 4, lines 30-58). *Borrill* teaches multiple ISA execution flags. A tag is

Art Unit: 2192

read for each of a plurality of instructions (see *Borrill*, col. 4, lines 30-62). Further, the function of the Dynamic Decode Unit (DDU) of *Borrill* is to translate non-native instructions (see *Borrill*, col. 5, lines 19-30). *Borrill* further teaches the instruction set architecture execution flags being set by a programming environment / operating system of the binary (see, for example, col. 2, line 58, through col. 3, line 11; and col. 4, lines 30-58; col. 3, lines 54-62 (operating system implementation)). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the method of *Yates et al.* to include such checking of settable controls (instruction set architecture execution flags) as per the teachings of *Borrill*. One would be motivated to do so to reduce overhead associated with translating and processing non-native instructions and facilitate easier incorporation of non-native instructions into executable code by assigning meaningful instruction identifiers recognizable by the native system as suggested by *Borrill* (see, e.g., *Borrill*, col. 2, line 58, through col. 3, line 11).

As per claim 34, the settable controls taught by *Borrill* control a level of semantic compatibility between the binary and the translated binary (they indicate the native ISA for “visiting” code; see, for example, col. 2, line 58, through col. 3, line 11; and col. 4, lines 30-58). Therefore, such a claim also would have been obvious for the same reasons stated above.

As per claim 36, the settable controls taught by *Borrill* are taught as being settable by a user (tags are inserted by a programmer; see, for example, col. 2, line 58, through col. 3, line 11). Therefore, such a claim also would have been obvious for the same reasons stated above.

As per claim 37, *Borrill* further teaches the translating and executing being based on a command, the instruction set architecture execution flags being set based on a number of command line flags associated with the command (see, for example, col. 2, line 58, through col.

Art Unit: 2192

3, line 11; and col. 4, lines 30-58). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to further modify the method of *Yates et al.* to include the setting of such command line flags as per the further teachings of *Borrill*. One would be motivated to do so to reduce overhead associated with translating and processing non-native instructions and facilitate easier incorporation of non-native instructions into executable code as suggested by *Borrill* (see, e.g., *Borrill*, col. 2, line 58, through col. 3, line 11).

As per claim 38, *Borrill* further teaches a register in a processor translating the binary being to store the instruction set architecture execution flags (see, for example, col. 4, lines 15-29). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to further modify the method of *Yates et al.* to include such storing/processing of architecture execution flags as per the further teachings of *Borrill*. One would be motivated to do so to as a necessary means of implementing and executing such instructions as suggested by *Borrill* (see, e.g., *Borrill*, col. 2, line 58, through col. 3, line 11; col. 4, lines 15-29).

As per claim 40, *Yates et al.* further disclose, in the case of self modifying code, the translating of the binary to include providing an instruction to controllers of memories that store the binary to perform write operations independent of whether the write operations modify a location where the binary is stored (see, for example, col. 10, lines 49-53). Therefore, for reasons stated above, such a claim also would have been obvious.

As per claim 41, *Yates et al.* further disclose the second instruction set architecture having an address space that is larger than the first instruction set architecture, the translating of the binary comprising using the address space of the second instruction set architecture (see, for

Art Unit: 2192

example, col. 83, lines 56-65). Therefore, for reasons stated above, such a claim also would have been obvious.

As per claim 42, in addition to the disclosure applied above to claim 33, *Yates et al.* fail to expressly disclose data accessed by the binary being stored in a single segment in memory and wherein an offset value for translating a virtual address to a physical address for the data is not modified during execution of the binary. However, *Borrill* teaches such handling of non-native addressing without modifying the non-native offset address (see, for example, col. 6, line 64, through col. 7, line 24). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the method of *Yates et al.* to include such non-native offset address handling as per the teachings of *Borrill*. One would be motivated to do so to provide simplified support and proper emulation for non-native instructions relying on a smaller address space as suggested by *Borrill* (see, e.g., *Borrill*, col. 6, line 64, through col. 7, line 24).

As per claim 43, this is a system claim substantially parallel to the claimed method discussed above (claims 30 and 31). *Yates et al.* further disclose a system (including a processor and DRAM) for implementing the prescribed methods (see, for example, Fig. 1; and col. 6, line 32, through col. 8, line 18), and all other limitations have been addressed as set forth above. Therefore, for reasons stated above, such a claim also would have been obvious.

As per claims 45, 47, and 48, these are system claims substantially parallel to the claimed methods discussed above (claims 38, 41, and 42, respectively). *Yates et al.* further disclose a system (including a processor and DRAM) for implementing the prescribed methods (see, for example, Fig. 1; and col. 6, line 32, through col. 8, line 18), and all other limitations have been

Art Unit: 2192

addressed as set forth above. Therefore, for reasons stated above, such claims also would have been obvious.

As per claims 49-55, these are apparatus claims substantially parallel to the claimed methods discussed above (see claims 38, 34, 35, 32, 40, 41, and 42, respectively). *Yates et al.* further disclose an apparatus for implementing the prescribed methods (see, for example, Fig. 1), and all other limitations have been addressed as set forth above. Therefore, for reasons stated above, such claims also would have been obvious.

As per claims 56-61, these are machine-readable medium claims substantially parallel to the claimed methods discussed above (see claims, 33, 34, 36-38, and 40 respectively). *Yates et al.* further disclose a machine-readable medium for implementing the prescribed methods (see, for example, Fig. 1), and all other limitations have been addressed as set forth above. Therefore, for reasons stated above, such claims also would have been obvious.

As per claims 62-65, these are system claims substantially parallel to the claimed methods discussed above (claims 40, 34, and 30). *Yates et al.* further disclose a system (including a processor and DRAM) for implementing the prescribed methods (see, for example, Fig. 1; and col. 6, line 32, through col. 8, line 18). *Yates* teaches that in the case of self-modifying code, these instructions are not translated because the execution is in general not predictable (*Yates*, col. 10, lines 44-53). However, the *Borrill* teachings further address the problems associated with self-modifying code, since all instructions are executed almost as native instructions are, by first being converted in real time by the DDU's, any code modifications are automatically accommodated, with no recompilation or other overhead encountered by binary translation or emulation (*Borrill*, col. 7, lines 25-32). Therefore, it would

Art Unit: 2192

have been obvious to one of ordinary skill in the art at the time the invention was made to modify the method of *Yates et al.* to include such automatic accommodation of self-modifying code (*i.e.*, with no need to additionally check whether the binary is self modifying) as per the teachings of *Borrill*. One would be motivated to do so to avoid the need for recompilation or other overhead by as suggested by *Borrill* (see, *e.g.*, *Borrill*, col. 7, lines 25-32).

As per claim 66, *Yates et al.* further disclose deviating from precise semantics of the binary during said translating in exchange for advantages offered by the second instruction set architecture (the original instructions are translated into different instructions for execution on a different platform which has the advantages of being, among other things, newer and faster; see, for example, col. 1, lines 50-60); and executing the translated binary (see, for example, the Abstract). Therefore, for reasons stated above, such a claim also would have been obvious.

7. Claim 39 is rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No 5,802,373 to Yates et al. in view of U.S. Patent No. 6,496,922 to Borrill and Bich C. Le, "An Out-of-Order Execution Technique for Runtime Binary Translators," 1998 (hereinafter *Le*).

As per claim 39, in addition to the disclosure applied above to claim 33, *Yates et al.* fail to expressly disclose the first instruction set architecture including in-order access to memory and the second instruction set architecture including out-of-order accesses to memory, the translating of the binary to include out-of-order accesses to memory by a processor executing the binary. However, *Le* teaches a runtime binary translator environment which facilitates out-of-order processing in the translated code (see, for example, sections 1.1 and 1.2). Therefore, it

Art Unit: 2192

would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the method of *Yates et al.* to include out-of-order accesses to memory as per the teachings of *Le*. One would be motivated to do so to achieve higher performance as suggested by *Le* (see, e.g., *Le* Abstract).

Conclusion

8. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

9. Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Eric B. Kiss whose telephone number is (571) 272-3699. The Examiner can normally be reached on Tue. - Fri., 7:00 am - 4:30 pm. The Examiner can also be reached on alternate Mondays.

Art Unit: 2192

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Tuan Dam, can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Any inquiry of a general nature should be directed to the TC 2100 Group receptionist: 571-272-2100.

EBK/EBK
November 20, 2006



TUAN DAM
SUPERVISORY PATENT EXAMINER